# Abgabe Übungsblatt 03

Technische Informatik 2 Wintersemester 2025/26

Baris Basar, Cengizhan Evcil, Nicolai Held

## Aufgabe 1

### a) Modul-Analyse mit nm (Funktion main)

Vorgehensweise: Die Objektdateien wurden mit dem Tool nm inspiziert, um die Symboltabelle zu prüfen. Symbole, die mit T (Text segment) markiert sind, sind in der jeweiligen Datei definiert.

Ergebnis:

| Modul | Relevanter Ausschnitt aus nm |
|---|---|
| stat.o | 0000000000000000 T main |
| mage.o | (Enthält keine Definition von main) |

Das Ergebnis von `nm stat.o`:

### b) Analyse der Linker-Fehler (undefined reference)

Analyse: Die Fehlermeldungen entstehen, weil eine Funktion, die von mage.o benötigt wird, im Linkvorgang nicht gefunden wird. Die nm-Analyse von mage.o zeigte die folgende undefinierte Referenz (U):

U _Z4initR15Initialisierung

Ursache: Dem Linker fehlt die Implementierung (Definition) der C++-Funktion:

void init(Initialisierung &)

Dieses Modul muss separat erstellt und hinzugefügt werden.

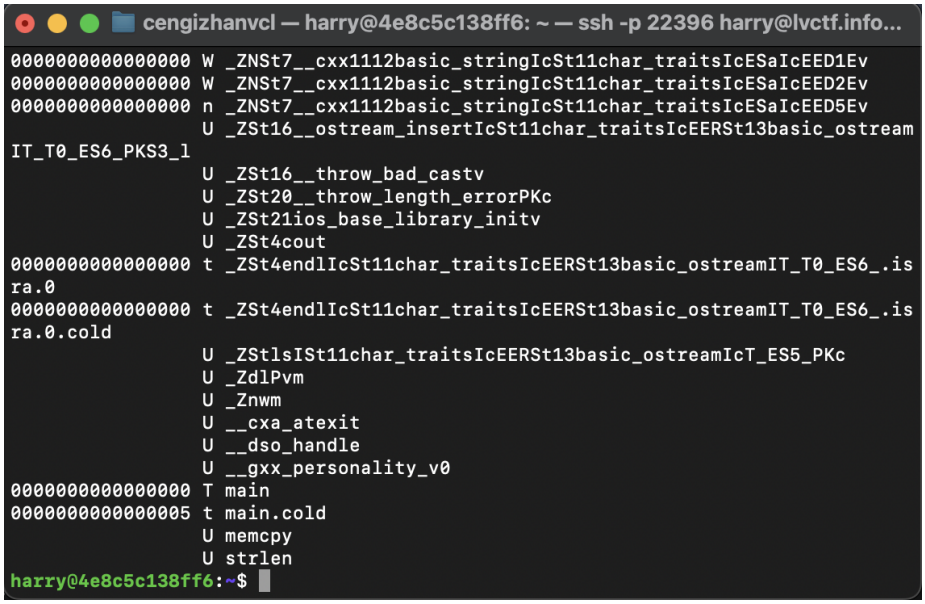


Figure 1: Kommando: nm stat.o

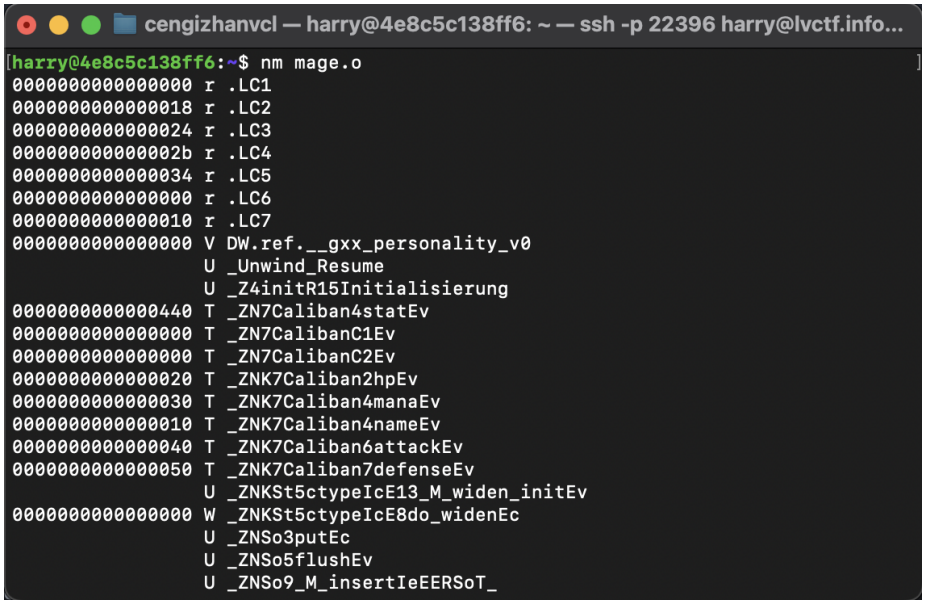


Figure 2: Kommando: nm mage.o

### c) Erstellung des Moduls values.cc

Ableitung des Namens: Die Symboltabelle von mage.o zeigte Klassenmethoden, die auf den Namen Caliban hindeuten (z.B. _ZN7CalibanC1Ev, _ZN7Caliban4statEv).

Implementation in values.cc:

Basierend auf den Definitionen in defs.hh (Makros HP, MANA, ATTACK, DEFENSE) und dem abgeleiteten Namen "Caliban" wurde die Initialisierungsfunktion implementiert:

```
#include "defs.hh"

void init(Initialisierung& data) {
    // Initialisierung der numerischen Felder mit den Werten aus defs.hh.
    data.hp = HP;
    data.mana = MANA;
    data.attack = ATTACK;
    data.defense = DEFENSE;

    // Setzen des Namens basierend auf der Symboltabelle
    data.name = "Caliban";
}
```

### d) Kompilieren, Linken und Ausführen des Programms

Ausgeführte Kommandos:

1. Kompilieren des neuen Moduls:

```
g++ -c values.cc -o values.o
```

2. Linken aller Module:

```
g++ mage.o stat.o values.o -o mage
```

3. Ausführen des Programms:

```
./mage
```

**Programmausgabe und Flagge:**

**Gefundene Flagge:**

flag21290e22-9689-45e1-bf20-421230aaada5

```
[harry@4e8c5c138ff6:~$ ./mage
Statistics of the great Caliban
==============================
Hitpoints: 728.9
Mana: 782
attack: 73, defense: 2
Flagge: flag{21290e22-9689-45e1-bf20-421230aaada5}
harry@4e8c5c138ff6:~$
```

Figure 3: Gefundene Flagge

| Priorität | Bearbeitungszeit |
|---|---|
| 1 | 2 |
| 2 | 1 |
| 3 | 4 |
| 4 | 2 |
| 5 | 3 |
| 6 | 2 |


Flag: flag{6bc73296-2059-4d7a-a53c-171842d8518c}

Figure 4: Flagge

Zeitpunkt 1: Aktuell behandelte Priorität: 0 Verloren gegangener Interrupt:

Zeitpunkt 2: Aktuell behandelte Priorität: 2 Verloren gegangener Interrupt:

Zeitpunkt 3: Aktuell behandelte Priorität: 0 Verloren gegangener Interrupt:

Zeitpunkt 4: Aktuell behandelte Priorität: 2 Verloren gegangener Interrupt:

Zeitpunkt 5: Aktuell behandelte Priorität: 6 Verloren gegangener Interrupt:

Zeitpunkt 6: Aktuell behandelte Priorität: 6 Verloren gegangener Interrupt:

Zeitpunkt 7: Aktuell behandelte Priorität: 4 Verloren gegangener Interrupt:

Zeitpunkt 8: Aktuell behandelte Priorität: 4 Verloren gegangener Interrupt:

Zeitpunkt 9: Aktuell behandelte Priorität: 5 Verloren gegangener Interrupt:

Zeitpunkt 10: Aktuell behandelte Priorität: 5 Verloren gegangener Interrupt:

Zeitpunkt 11: Aktuell behandelte Priorität: 5 Verloren gegangener Interrupt: 5

Zeitpunkt 12: Aktuell behandelte Priorität: 3 Verloren gegangener Interrupt:

Zeitpunkt 13: Aktuell behandelte Priorität: 3 Verloren gegangener Interrupt:

Zeitpunkt 14: Aktuell behandelte Priorität: 3 Verloren gegangener Interrupt:

Zeitpunkt 15: Aktuell behandelte Priorität: 3 Verloren gegangener Interrupt:

Zeitpunkt 16: Aktuell behandelte Priorität: 2 Verloren gegangener Interrupt:

Zeitpunkt 17: Aktuell behandelte Priorität: 1 Verloren gegangener Interrupt:

Zeitpunkt 18: Aktuell behandelte Priorität: 1 Verloren gegangener Interrupt:

Zeitpunkt 19: Aktuell behandelte Priorität: 0 Verloren gegangener Interrupt:

Zeitpunkt 20: Aktuell behandelte Priorität: 0 Verloren gegangener Interrupt:

Zeitpunkt 21: Aktuell behandelte Priorität: 0 Verloren gegangener Interrupt:

Absenden

Figure 5: Loesung

# Aufgabe 2

## a. Die Flagge & die Lösung

Wie in der Aufgabenstellung[1] erwähnt, sind Prozesse "zustandsorientiert" und gleiche Interrupts können zu einem Zeitpunkt nur einmal vorliegen. Daher wird der *zweite Interrupt mit der Priorität 5* vernachlässigt und geht *verloren*, denn zu jenem Zeitpunkt wird bereits ein Interrupt mit Priorität 5 bearbeitet. Einige Interrupts werden eingereiht, da höherwertig priorisierte Interrupts anstehen[1](*Priority Scheduling*) (je größer die Priorität desto dringlicher), dies wird hoffentlich in der folgenden Auflistung leicht ersichtlich: (Die Bearbeitungsdauer wird in den Klammern angegegeben)

| | Aktuell: | Eingreiht: | Verloren: |
|---|---|---|---|
| Zeitpunkt 1: | 0 | | |
| Zeitpunkt 2: | 2(1) | | |
| Zeitpunkt 3: | 0 | | |
| Zeitpunkt 4: | 2(1) | | |
| Zeitpunkt 5: | 6(2) | | |
| Zeitpunkt 6: | 6(1) | 4(2) | |
| Zeitpunkt 7: | 4(2) | 2(1) | |
| Zeitpunkt 8: | 4(1) | 2(1) | |
| Zeitpunkt 9: | 5(3) | 2(1) | |
| Zeitpunkt10: | 5(2) | 2(1) | |
| Zeitpunkt11: | 5(1) | 2(1) | 5(3) |
| Zeitpunkt12: | 3(4) | 2(1) | |
| Zeitpunkt13: | 3(3) | 2(1),1(2) | |
| Zeitpunkt14: | 3(2) | 2(1),1(2) | |
| Zeitpunkt15: | 3(1) | 2(1),1(2) | |
| Zeitpunkt16: | 2(1) | 1(2) | |
| Zeitpunkt17: | 1(2) | | |
| Zeitpunkt18: | 1(1) | | |
| Zeitpunkt19: | 0 | | |
| Zeitpunkt20: | 0 | | |
| Zeitpunkt21: | 0 | | |

Figure 6: Auflistung

Kein *eingereihter* Interrupt meldet sich nochmal(bevor er fertig bearbeitet wird), daher geht auch kein *eingreihter* Interrupt verloren. Zum *Zeitpunkt 7* wird ein *Interrupt mit Priorität 2* eingereiht und dieser kann erst zum *Zeitpunkt 16* beabreitet werden, da bis dahin erst höherwertig priorisierte Interrupts bearbeitet werden. Zu *Zeitpunkt 13* meldet sich bereits der *Interrupt mit Priorität 1*, dieser muss sich aber aufgrund des *Interrupts mit Priorität 2* hinter diesem einreihen und bis *Zeitpunkt 17* gedulden.

**b) Ablauf**

Der resultierende Ablauf ist zufriedenstellend, falls der verlorene Interrupt mit der Priorität 5 keine kritischen Informationen (auch ein Zeitstempel kann je nach Umstand kritisch sein) bearbeiten wollte. Da die Interrupts aber zustandsorientiert sind, also eine mehrfache gleichzeitige Einreihung nicht erlaubt ist, und diese Entscheidung wohl bewusst bei dem System getroffen wurde, lässt sich davon ausgehen, dass dieser verlorene Interrupt nicht kritisch war und so sogar Effizienz gewonnen wurde, sodass sich die Bearbeitung anderer Interrupts nicht noch weiter aufschieben konnte.

## Literaturverzeichnis


1) OpenAI ChatGPT Chatverlauf (s. Anhang) (11.11.2025)
2) Aufgabenstellung der Challenge "Unterbrechungen" auf der CTF-Plattform der Uni-Bremen des Moduls TI2 WS25/26 10.11.25 (Screenshot siehe Anhang)


# Anhang

1) Chatverlauf mit [ChatGPT] (https://chat.openai.com/chat) am 04.11.2025: Frage: Was macht nm Kommando in Unix Weitere Frage: Kompilieren, Linken und Ausführen Kommandos

nm liest Objektdateien (.o), Bibliotheken (.a, .so) oder ausführbare Dateien und zeigt die Symboltabelle an — also alle Funktionen, Variablen, und deren Speicheradressen.

| Schritt | Befehl | Beschreibung |
|---|---|---|
| 1 Kompilieren | `gcc -c file.c -o file.o` | Übersetzt in Objektdatei |
| 2 Symbolanalyse | `nm file.o` | Zeigt Symbole (Funktionen, Variablen) |
| 3 Linken | `gcc file.o -o program` | Erstellt ausführbare Datei |
| 4 Prüfen der Abhängigkeiten | `ldd ./program` | Zeigt benötigte Bibliotheken |
| 5 Ausführen | `./program` | Startet das Programm |
| 6 Debugging | `gdb ./program` | Startet Debugger |

2)


## Unterbrechungen

In einem System werden für einen gewissen Zeitraum die eingehenden Unterbrechungen (Interrupts) der verschiedenen Prioritätsstufen (je höher, desto dringender) beobachtet.

Wie würden die folgenden Unterbrechungen unter Berücksichtigung der in der Tabelle aufgeführten Bearbeitungszeiten abgearbeitet? Die Interrupts sind zustandsorientiert, gleiche Interrupts können zu jeden Zeitpunkt nur einmal anliegen. Wenn kein Interrupt bearbeitet wird, ist der gegenwärtig ausgeführte Prozess an der Reihe (Priorität 0).

Figure 7: Aufgabenstellung CTF-Plattform Uni-Bremen TI2 WS25/26 - Unterbrechungen